FYS-STK4155 - Project 2

Geir Tore Ulvik, Idun Klvstad

November 22, 2018

Abstract

In this project we explore classification and regression applied to the well-known Ising model. We show that the Ising model can be well represented by linear regression methods like ordinary least squares, Ridge, and LASSO (R2 score of 1.0 for all methods). Moving on the the two-dimensional Ising model, we show that in classifying the states as either ordered or disordered, the logistic regressor is mediocre at best (under 65% correctly classified states).

Classifcation with both single- and multilayer perceptron is show to be very successful in labling the input states, even for states in the critical temperature region (up to 97%)). Moreover, the neural network needs far less data for training than the logistic regressor for fitting.

1 Introduction

Code, data and figures are available at the following GitHub address: GitHub repository
This project will follow closely the article of Metha et al, arXiv 1803.08823. The article is listed in references: [1].

The main aim of the project is to study both classification and regression problems. The project will include three regression algorithms, logistic regression for classification problems and a multilayer perceptron code. The multilayer perceptron code is for studying both regression and classification problems.

We will use the so-called Ising model for the training data, and also focus on supervised learning.

As mentioned above we will look at both regression and classification problems. Regression will be used to determine the value of the coupling constant of the energy of the one-dimensional Ising model. The classification case will study the one dimensional Ising model in order to classify the phase of the model.

The various algorithms will be seen in comparison. In the end we will summarize and give a critical evaluation of pros and cons. We will look at which algorithm works best for the regression case, and which works best for the classification case. [2]

2 Theory

As the theory behond the three regression methods used as well as error estimates and the bootstrap method is covered in the previous project, we will not restate it here. Alternatively, see [1] and/or [3].

2.1 The Ising model

The ising model is a simple binary value system where the variables in the model can take only two values. For examle ± 1 or 0 and 1. [2]

We will look at the physicist's approach, and call the variables for spin. [2]

Given an ensamble of random spin configurations we can assign an energy to each state, using the 1D Ising model with nearest-neighbor interactions:

$$E = -J \sum_{j=1}^{N} S_j S_{j+1} \tag{1}$$

J is the nearest-neighbor spin interaction, and $S_j \epsilon \pm 1$ is a spin variable. N is the chain length. [1] [2]

In one dimension, this model has no phase transitions at finite temperature. [2]

To get a spin model with pairwise interactions between every pair of variables, we choose the following model class:

$$E_{model}[S^i] = -\sum_{j=1}^{N} \sum_{k=1}^{N} J_{j,k} S_j^i S_k^i \tag{2}$$

[1]

In this equation i represents a particular spin configuration. [2]

The goal with this model is to determine the interaction matrix $J_{j,k}$. As the model is linear in $\mathbf{J}$, it is possible to use linear regression.

The problem can be recast on the form

$$E_{model}^i = \mathbf{x}^i \cdot \mathbf{J} \tag{3}$$

2.2 Logistic regression and classification problems

Differently to linear regression, classification problems are concerned with outcomes taking the form of discrete variables. For a specific physical problem, we'd like to identify its state, say whether it is an ordered of disordered system. [3]

Logistic regression can be used to define the phases of the Ising model. [**?**]

Configurations representing states below the critical temperature are called ordered states, while those above the critical temperature are called disorderes states. [2]

The theoretical critical temperature for a phase transition is $T_C \approx 2.269$.

2.3 Cost functions

In order for a network and a logistic regressor to improve it needs a way to track how it's performing. This is the purpose of a cost function. Essentially, the cost function says something about how wrong the model is in classifying the input. The objective in machine learning, and logistic regression, is then to minimize this error.

The cost function used in this project is called the **cross-entropy**, or the 'negative log likelihood', and takes the form

$$\mathcal{C}(\hat{\beta}) = -\sum_{i=1}^{n} \left(y_i(\beta_0 + \beta_1 x_i) - \log\left(1 + \exp\left(\beta_0 + \beta_1 x_i\right)\right)\right) \tag{4}$$

2.4 Gradient Descent

Minimizing the cost function is done using Gradient Descent. The jist of it is that in order to optimize the weights or coefficients, and biases to minimize the cost function, one can change their values to

$$\frac{\partial \mathcal{C}(\hat{\beta})}{\partial \hat{\beta}} = -\hat{X}^T\left(\hat{y} - \hat{p}\right) \tag{5}$$

2.4.1 Stochastic gradient decent

The stochastic gradient decent method address some of the shortcomings of the normal gradient decent method. The gradient decent method is for instance sensitive to the choise of learning rate [3].

The underlying idea of stochastic gradient decent comes form observing that the cost function we want to minimize, almost always can be written as a sum over n data points. [3]. Which gives

$$C(\beta) = \sum_{i=1}^{n} c_i(\mathbf{x}_i\beta) \tag{6}$$

[3]

This means that we also can find the gradient as a sum over i gradients as follows:

$$\Delta_\beta C(\beta) = \sum_{i}^{n} \Delta_\beta c_i(\mathbf{x}_i\beta) \tag{7}$$

[3]

Randomness is included by only taking the gradient on a subset of data.

2.5 Accuracy score

To measure the accuracy of the network, an accuracy score is defined as:

$$\text{Accuracy} = \frac{\sum_{i=1}^{n} I(t_i = y_i)}{n}$$

which is simply the number of correctly labeled states divided by the total number of states. In the equation above, I is the indicator function, 1 if $t_i = y_i$ and 0 otherwise. t_i is the known correct label and y_i is the label output by the network.

2.6 Neural Network

The concept of a neural network is essentially to mimick how neurons in the brain are connected and learn. The network consists of interconnected layers of neurons, or 'nodes'. In the type of network used in this project, a Feed Forward Neural Network (FFNN), each node in a layer has a connection to every single node in the previous layer. As an input signal is 'fed' forward through the network, each node in each layer will 'fire' or 'activate' based on the sum of the signals from the nodes in the preceding layer, until the signal reaches the output layer which, in this project, outputs a probability that the original input is in one of two classes. In short, the FFNN is a binary classifier which takes an input and outputs the likelihood of that input belonging to one of the classes.

The network doesn't know how to do this without first being trained. Training the network involves analyzing the error the network makes in classifying an input, and propagating this error backwards through the network such that the next time an input of that class is seen, the network will be better at classifying it. An example of this type of binary classification could be to say whether or not there is a cat in a picture.

Numerous articles and books are available on the subject of neural networks, so for an in-depth explanation of the concepts involved (especially the backpropagation of error), we recommend seeking out these texts. Examples are the online book Nielsen's, or the article by Mehta et al ([1]).

3 Method

3.1 Producing data and recasting problem

To generate the training data, we used the code given in lecture notes, same as the notebooks from Mehta et al. The code generates a system with size $L = 40$ and 10000 different ising states and returns the Ising-energies.

Then the problem was recast as a linear regression model, using the regression methods from project1. The theory behind the Ising model and the recasting is described in 2.1.

3.2 Determining the phase of the 2D Ising model

To determine the phase of the two dimensional Ising model we use logistic regression. Information about logistic regression can be found in section 2.2.

Out logistic regression method uses a cost function and gradient decent, which you can read about in sections 2.3 and 2.4.

We chose to use a stocastic gradient decent method.

We use the data sets generated by Mehta et al [1] and a fixed lattice of L x L = 40 x 40 spins in two dimensions.

To evaluate the model, we have used the accuracy score described in section 2.5.

3.3 Classifying with Neural Network

The codebase provided in (cite neural network lecture notes) was used as a base. However, the network was implemented using the sigmoid activation function in the output layer as well, making the network 'tailored' to binary classification (only one output node). The following code snippet is the bread and butter of the neural network. Weights are initialized to random values following a normal distribution (`numpy.random.randn`), and biases to the low value of 0.01.

```
def train(self):
    indices = np.arange(self.n_inputs)
    for i in range(self.epochs):
        for j in range(self.n_iter):
            chosen_indices = np.random.choice(
                indices, size=self.batch_size, replace=False)
            # Batch training data and targets
            self.x_batch = self.x[chosen_indices]
            self.y_batch = self.y[chosen_indices]
            activations = self.feed_forward(self.x_batch)
            self.backwards_propagation(activations)
```

4 Results

4.1 Linear Regression

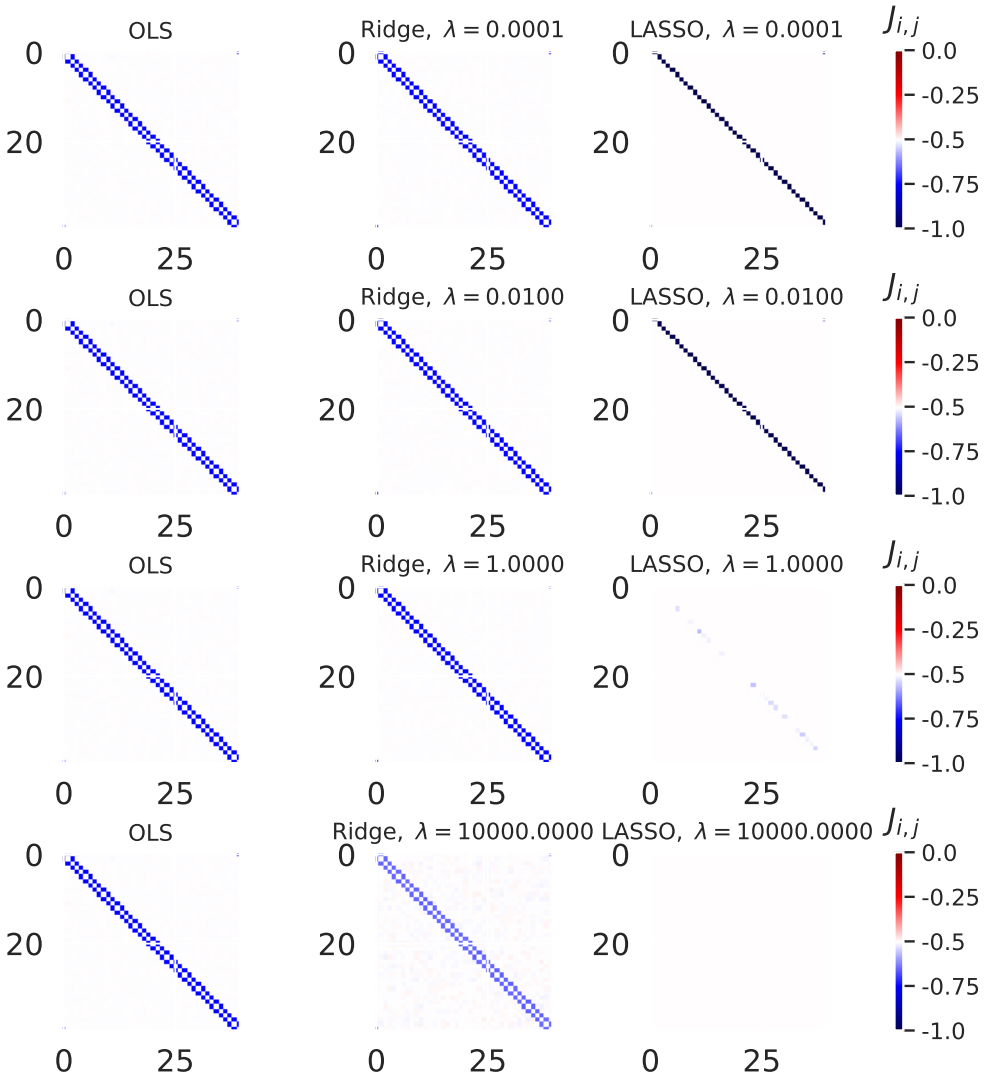


Figure 1: Learned interaction matrix $J_{i,j}$ for the Ising model, for select regularization strengths λ, generated based on our regression models. OLS is not dependent on *ambda*, but shown for comparison.

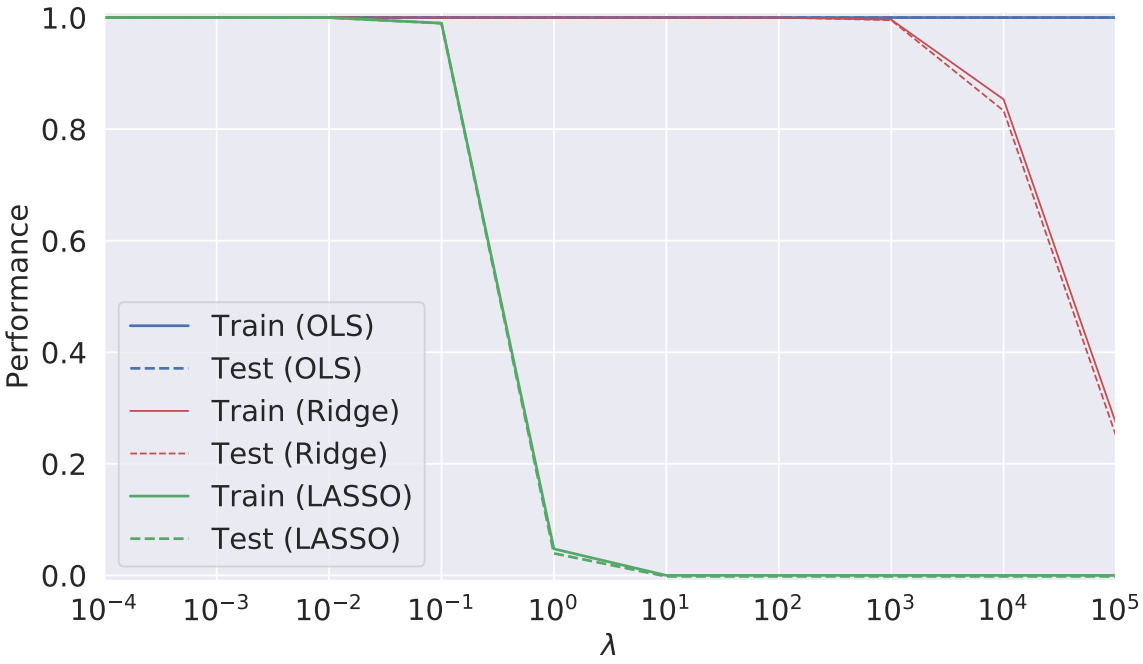


Figure 2: R2 score performance of the linear regression models as a function of regression parameter λ.

Figure 2 shows how the R2 score varies bestween models. It's important to note that the λ for Ridge regression, and α for Lasso regression have the same value, but they affect the models on different orders of magnitude, and must be treated somewhat separately. Even so, we can see that the training and test set R2-scores follow eachother closely.

Figure 6 is from article [1] and shows performance of OLS, Ridge and LASSO regression on the Ising model as measured by the R2 score.

4.2 Classifying with Logistic Regression

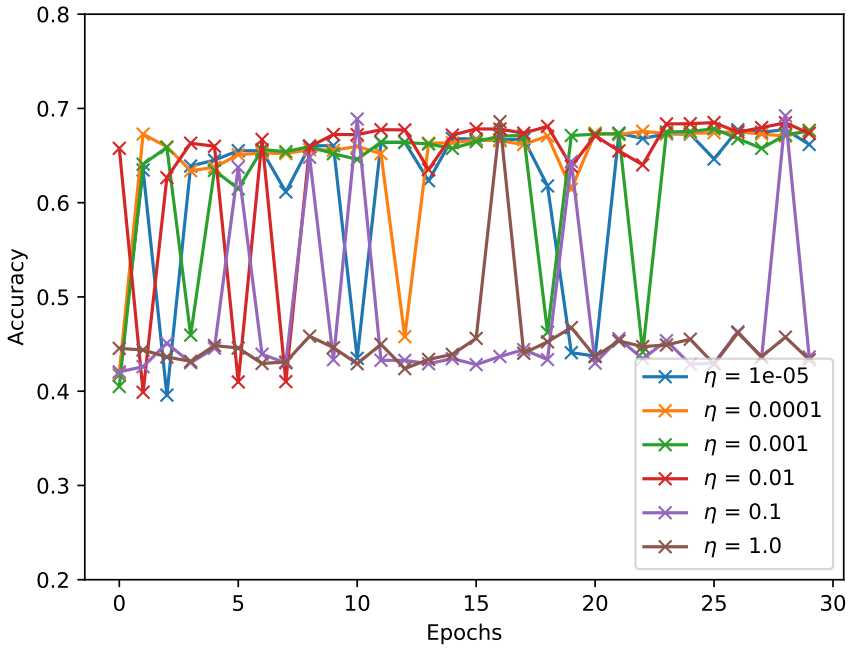


Figure 3: Accuracies for a selection of learning rates η as a function of epochs. 30 epochs, batch size = 100, and momentum parameter $\gamma = 0.01$. The accuracy values are measured on the test set. The ratio of amount of training data to test data is 0.8.

η	Training	Test	Critical
10^{-5}	0.718	0.680	0.616
10^{-4}	0.723	0.683	0.624
10^{-3}	0.723	0.685	0.628
10^{-2}	0.712	0.672	0.604
10^{-1}	0.462	0.430	0.460
1	0.465	0.446	0.480

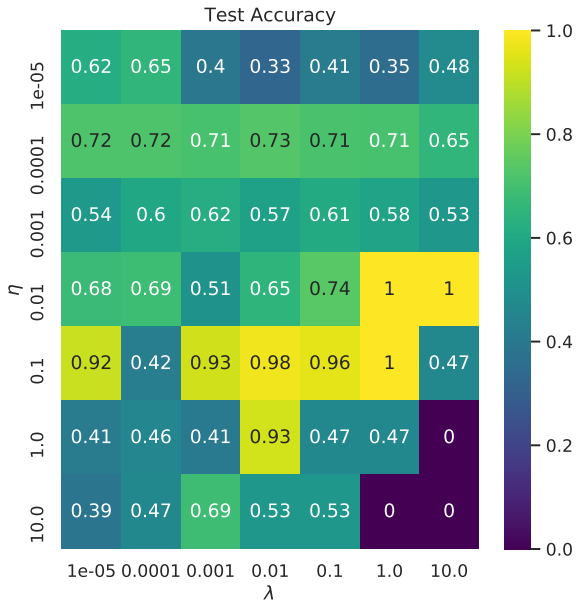
Table 1: Accuracies for a selection of learning rates η after 30 epochs. Batch size = 100, and momentum parameter $\gamma = 0.01$. The ratio of amount of training data to test data is 0.8

In table 1 the accuracy on data containing critical states is included. Due to the varying nature our logistic model, a solution in which the weights producing the best fit on test data are stored was developed. The accuracies for the critical states are thus not necessarily

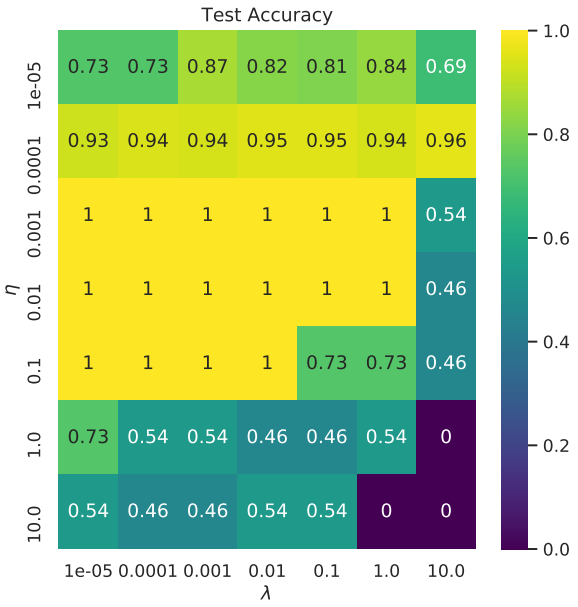
produced with weights as they are after 30 epochs. The reason for this descrepancy is yet to be found.

4.3 Classifying with Neural Network

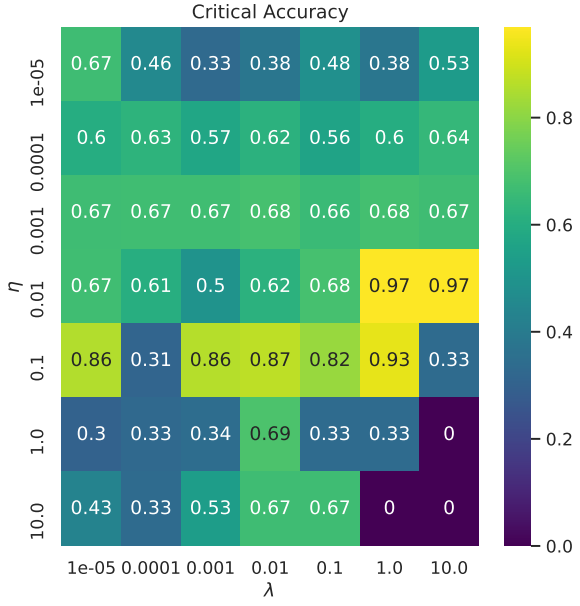
Grid search across multiple orders of magnitude of learning rate and regularization parameter is shown in figure 4. Both network architectures were trained on 10% of the available samples from the ordered and disordered states. Comparing the values with those presented in table 1, the network performs much better in classifying the states. The grid search shows that for the test set the larger architecture with two layers perform very well on the test set across multiple combinations of parameters, while the single-layer is more sensitive to the parameter choice. The performance on the test set is, however, not reflected in the two architectures' performance on the data from the critial temperature region, as seen in figure 4 (c,d), where the single-layer network actually outperforms the multilayer network slightly.



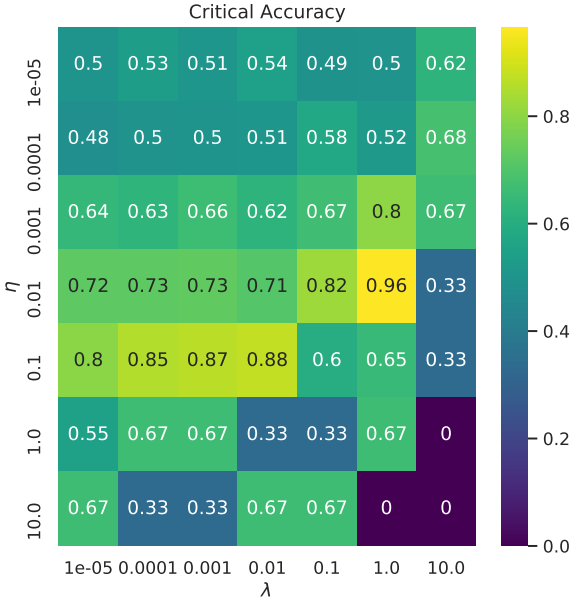
(a)



(b)



(c)



(d)

Figure 4: Neural network classification accuracies for grid search across learning rates η and regularization parameters λ on the test and critical set after 10 epochs. a), c) show the network with one hidden layer, 10 nodes. b), d) show the network with two hidden layers of size 100 (first) and 50 (second).

5 Discussion

5.1 Linear Regression

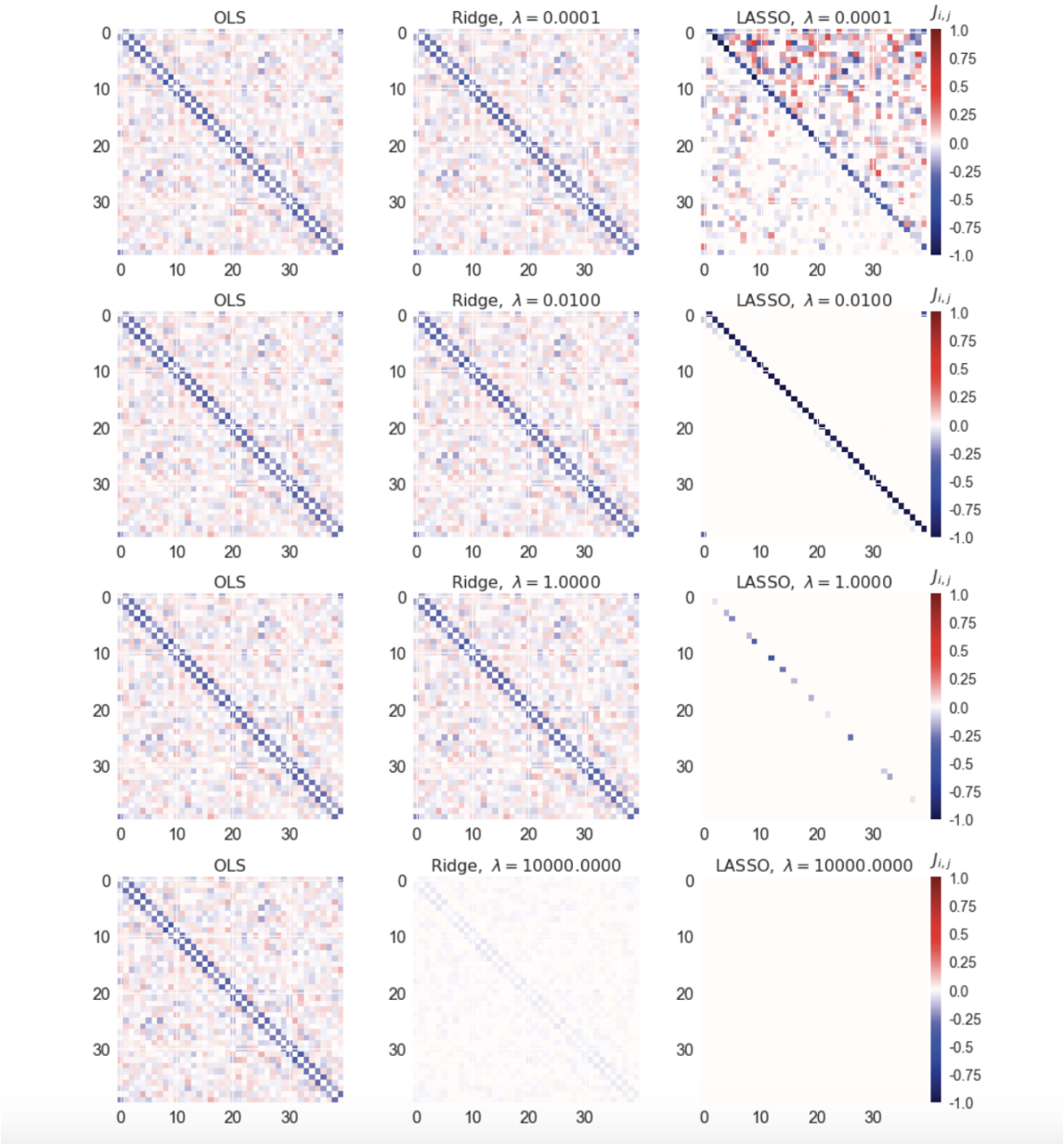


Figure 5: Figures from article [1] shows similarities to the ones generated by our regression models.

Figures 1 and 5 show that both regression models succeed in producing interaction matrices with most values along the diagonal. However, the much larger amount of training data

used for our runs (the full 10000 samples) give more accurate results, as can be seen from the lack of values outside the diagonals in the interaction matrices. Reducing the number of samples to 400 produces results of very similar character to those of Mehta et al.

Looking at the R2 scores from figure 2, we see that the R2 score for the training and test set follow each other closely. Again, comparing with articles plot, shown in figure 6, the two have a similar shape but there are a larger difference between the R2 score for training data and the test data in the article. In the articles' figure, the difference seems to be biggest for Ridge and OLS. As for our plot, it seems like Ridge has a bit bigger difference between the R2 score for training and test data than for the two other methods. Again this can likely be attributed to the amount of data regression was performed on.

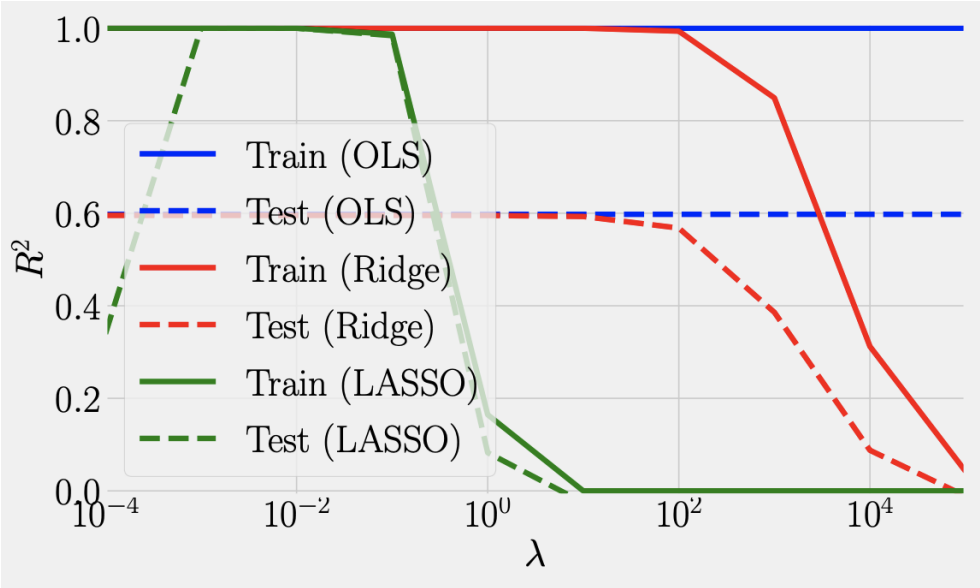


Figure 6: For comparison: Performance of OLS, Ridge and LASSO regression on the 1D Ising model as measured by the R2 score. Figure 16 in [1]

5.2 Classifying with Logistic Regression

In figure 3, it is apparent that for some η the model quickly rises to an approximate highest value, but jumps back down to the equivalent of guessing. When approaching 30 epochs and more, this behaviour diminishes somewhat, and overall the etas that produce the best results ($10^{-5} - 10^{-2}$) have most of their values in the higher points. From table 1 we see that the accuracy is best for $\eta = 10^{-3}$. Even so, the accuracy is still way below 1.0, making this logistic model only marginally better than guessing. Table 1 can be compared to figure 7. We see that our model is slightly worse than the one described in the article, but that they follow a similar trend.

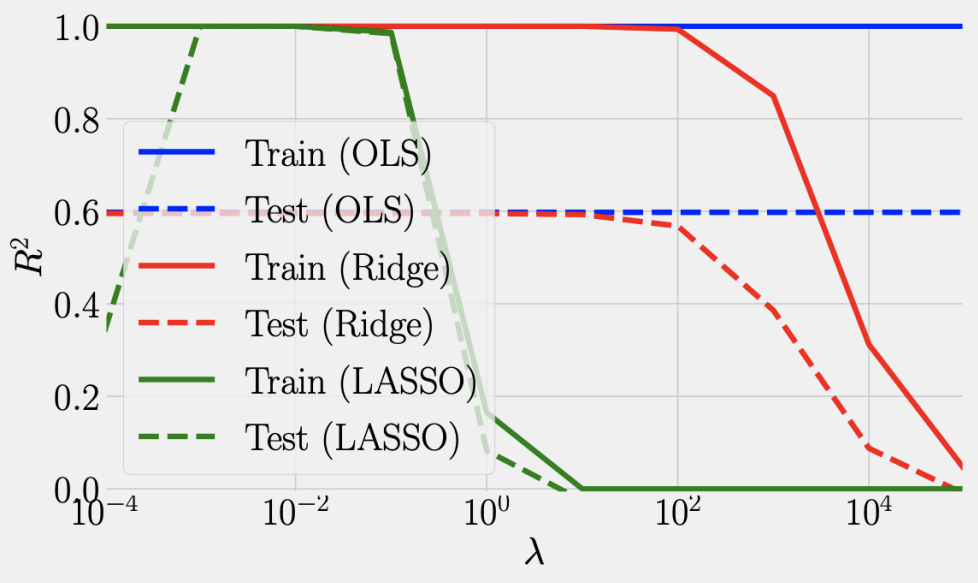


Figure 7: Accuracy as a function of the regularization parameter λ in classifying the phases of the 2D Ising model on the training (blue), test (red), and critical (green) data. The solid and dashed lines compare the liblinear and SGD solvers, respectively..Figure 21 in [1]

5.3 Classifying with Neural Network

The neural network results in figure 4 show that the network performs far better than logistic regression, reaching 1.0 accuracy on both training and test sets, and as much as 0.97 on the critical set. Adding that the network is trained on only 10% of the data that the logistic regressor was applied to, it is significantly better, and faster. Note that the network with a more complex architecture, with two layers and far more nodes in each layer, doesn't perform better on the data from the critical temperature region. This indicates that performance on the test set alone is not a good indicator of how well the network generalizes. From the grid searches, it also seems that the regularization parameter plays a larger role for the less complex network.

The fact that the network gains practically nothing in terms of generalization, when increasing the amount of layers and nodes likely indicates that the problem is not a very complex one. Thus, two transformations and 10 nodes is all the network needs in order to quite accurately classify the states.

6 Conclusion

In this project we have explored classification and regression applied to the well-known Ising model. We showed that the ising model can be well represented by linear regression methods like ordinary least squares, Ridge, and LASSO. Moving on the the two-dimensional Ising model, we show that in classifying the states as either ordered or disordered, the logistic regressor is mediocre at best. However, the logistic regressor in this project does not utilize a regularization term (rather a momentum parameter). Implementing that may change the outcome somewhat, but overfitting of the data does not seem to be the issue.

Classifcation with both single- and multilayer perceptron is show to be very successful in labling the input states, even for states in the critical temperature region. Moreover, the neural network needs far less data for training than the logistic regressor for fitting.

As a binary classifier, the current implementation is fine, but it can be expanded to function well for classifying data with more than two classes, for example by applying the softmax function in the output layer, using one node for each class. Then again, with packages like AutoKeras around, it is more suitable as a learning exercise than for actual production use.

References

[1] Pankaj Metha et al. A high-bias, low-variance introduction to machine learning for physicists. 03 2018.

[2] University of Oslo Department of Physics. Project 2 on machine learning. https://compphysics.github.io/MachineLearning/doc/Projects/2018/Project2/pdf/Project2.pdf, 2018.

[3] M Hjorth-Jensen. Data analysis and machine learning: Linear regression and more advanced regression analysis - lecture notes. https://compphysics.github.io/MachineLearning/doc/pub/Regression/html/._Regression-bs000.html, 2018.